Class: S7 ICT

Date: Tuesday, April the 18th, 2023

Teacher: Mr Barsamian

B Test — With computer

Family name: _____

First name: _____

Grade: ___ / 10

Duration: 1 hour (60 minutes).

This test has to be done both on paper and on computer. At the end of the test, make sure to upload the python file on your teacher's USB key.

Some questions are bonuses, and it is highly advised to do them only at the end, when everything else has been done.

If needed, the candidate can also handle some comments inside the code or on paper.

Please keep track of the clock, and avoid spending too much time on a question. Stay focused, and good luck!

Short description of this work:

We will handle strings using a Tree data structure. The start of a string is the root of the Tree, and then we will figure out different ways to encode them.

1 Introduction — 0 points

Please start by downloading the following file, that you'll have to update for this test:

http://www.barsamian.am/2022-2023/S7ICTB/BTest_Words.py

A string consists of multiple characters. We will model the strings seen in this test by trees, see Listing 1. If a tree models a string, then each of its nodes will either be empty, either will contain a single character `data` and have two children `left` and `right`.

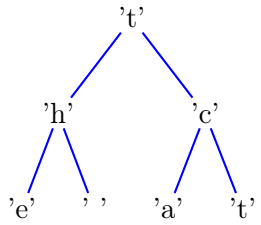
```
1  class Tree:
2      def __init__(self, data, left=None, right=None):
3          self.data = data
4          self.left = left
5          self.right = right
6
7      def __str__(self):
8          return str(self.data)
```

Listing 1: Python code for the Tree data structure.

The file contains Listing 2, which implements the tree on the right.

```
1  str1_node4 = Tree('e')
2  str1_node5 = Tree(' ')
3  str1_node6 = Tree('a')
4  str1_node7 = Tree('t')
5  str1_node2 = Tree('h', str1_node4, str1_node5)
6  str1_node3 = Tree('c', str1_node6, str1_node7)
7  str1_node1 = Tree('t', str1_node2, str1_node3)
```

Listing 2: Python code for the first string.



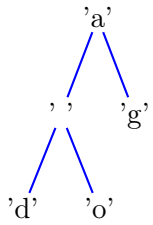
2 First steps — 5 points

1. The file also contains Listing 3. Please draw, on paper, a tree which would correspond to this implementation.

```
str2_node8 = Tree('s')
str2_node9 = Tree(' ')
str2_node10 = Tree('s')
str2_node11 = Tree(' ')
str2_node4 = Tree('i', str2_node8, str2_node9)
str2_node5 = Tree('i', str2_node10, str2_node11)
str2_node6 = Tree('t')
str2_node7 = Tree('!')
str2_node2 = Tree('h', str2_node4, str2_node5)
str2_node3 = Tree('i', str2_node6, str2_node7)
str2_node1 = Tree('t', str2_node2, str2_node3)
```

Listing 3: Python code for the second string.

2. Let us consider the tree drawn below. Please write, in the Python file, a Tree which would model this string.



3. In the Python file, the functions `mystery1`, `mystery2` and `mystery3` are provided (also given in Listing 4) which take as argument a tree. Please:
   (a) explain step by step what happens if you execute `mystery1(str1_node3)`;
   (b) explain step by step what happens if you execute `mystery2(str1_node3)`;
   (c) explain step by step what happens if you execute `mystery3(str1_node3)`.

```
def mystery1(tree):
    if tree == None:
        return ""
    return mystery1(tree.left) + mystery1(tree.right) + tree.data

def mystery2(tree):
    if tree == None:
        return ""
    return mystery2(tree.left) + tree.data + mystery2(tree.right)

def mystery3(tree):
    if tree == None:
        return ""
    return tree.data + mystery3(tree.left) + mystery3(tree.right)
```

Listing 4: Python code for the mystery functions.

3 Counting — 5 points

1. Please give a function `count_nodes` that takes as argument a tree and that counts the number of nodes in that tree. We only count the nodes that are drawn on the representations on the previous pages, we don't count `None`.

   Unit tests:

   - `count_nodes(str1_node1)` should return 7;
   - `count_nodes(str2_node1)` should return 11.

2. Please give a function `count_spaces` that takes as argument a tree and that counts the number of spaces in that tree.

   Unit tests:

   - `count_spaces(str1_node1)` should return 1;
   - `count_spaces(str2_node1)` should return 2.

3. We now define the function `string_representation` which is the function `mystery3`. This means that the string associated to a tree is the result of the function `mystery3` on that tree. Use the function `count_spaces` to create a new function `count_words` that takes as argument a tree and that counts the number of words in the string associated to this tree.

   Unit tests:

   - `count_words(str1_node1)` should return 2;
   - `count_words(str2_node1)` should return 3.

4. A tree is correctly representing a string if all its nodes have a single character in the `data` field. All trees given previously are correct, but for instance the tree defined by `Tree('the cat')` is not correct.

   Please write a function `is_correct` that takes as argument a tree and that returns True if this tree is correct, and that returns False otherwise.

   BONUS — The data field is not necessarily a string. Verify that your function works is, for instance, the data field contains a number.

BONUS Please write a function `is_really_correct` that takes as argument a tree and checks that the root node contains a capital letter (uppercase), and that all other nodes contain either a space, either a letter (lowercase or uppercase), either punctuation marks (`! ? , ; : - ( ) " ' .`).